Lecture 5

Mathematical tools for image processing, continued

Spatial versus transform operations

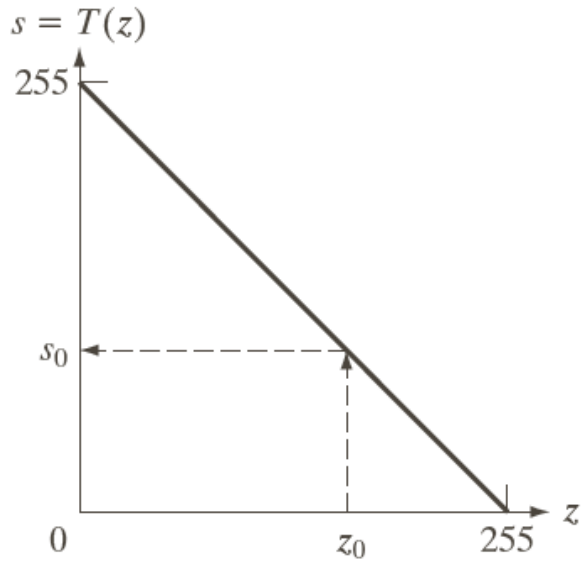
- Spatial operations are performed in spatial domain
  - Image is represented as f(x,y)
  - X=0,1,2,…,M
  - Y=0,1,2,…,N
  - Single pixel based operations, neighborhood operations, geometric transformation
- Transform operation
  - Image is transformed to other domain g(u,v) (Transform domain)
  - Operations are performed in the other domain
  - Transformed operated image is inverse-transformed to the spatial domain

Single pixel spatial operations

Intensity mapping
s=T(z)
z: input image intensity
s : Output image intensity
T : the operation

Example: getting the negative of 8 bit image

T(z)= 2^8-z



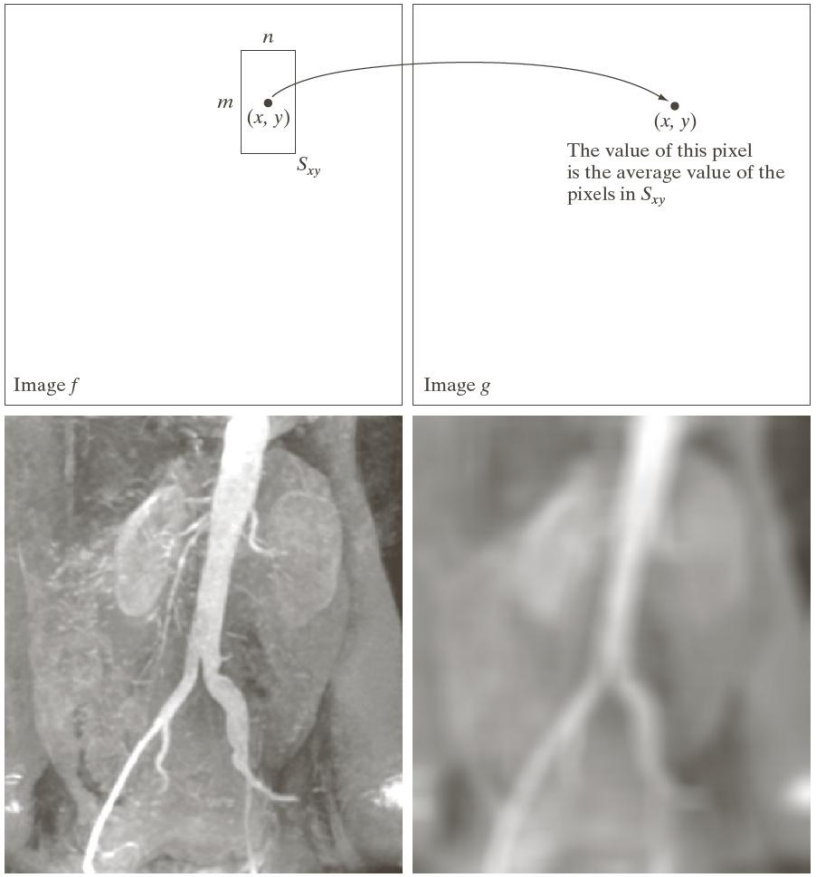
Neighborhood spatial operations

The value of the pixel at x,y in the output image is calculated from a set of neighbor pixels centered at x,y in the input image

Example, the calculated value = the average of the neighbors rectangle of dimensions M by N centered at the pixel

$$g(x,y) = \frac{1}{MN} \sum_{(r,c)\in S_{x,y}} f(r,c)$$

r, c are row and column index in the M xN neighbor set S



Geometric spatial transformation

Geometric spatial transformation consists of two basic operations: 1)spatial transformation of coordinates.
2) Intensity interpolation

- Geometric transformation is the mapping of the coordinates of each pixel in an input image to another (displaced/rotated,..) pixel in the output image.
- Intensity interpolation is used to assign intensities for the relocated pixels in processed image
- Forward mapping: for each pixel in the input image, find its location in the output image and assign its value
  - Multiple output values are assigned to the same output pixel (Overlaps)
  - Some output locations may not assigned values (Holes)
- Inverse mapping: for each pixel in the output image, find the location in the input image by applying the inverse transform, use the interpolation to calculate intensity value based on the intensities in the input image location

  $(u,v)= T^{-1}(x,y)$

  Inverse mapping avoids holes and overlaps in the output image. So it is more efficient than forward mapping and preffered in geometric transformation applications(such as Matlab).

u,v are pixel coordinates in the original image,while x,y are the corresponding pixel coordinates in the transformed image

$$(x,y) = T\{(u,v)\}$$

$$(x,y) = \left(\frac{u}{2}, \frac{v}{2}\right) \text{(shrinking)}$$

Affine tranformation

$$[x, y, 1] = [v, w, 1]T$$

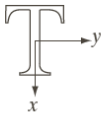
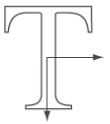
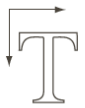
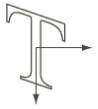
$$= [v, w, 1]\begin{bmatrix} t_{11} & t_{12} & 0 \\ t_{21} & t_{22} & 0 \\ t_{31} & t_{32} & 1 \end{bmatrix}$$

T is the transormation matrix

This transformation can scale, rotate, translate a set of pixels, depending on the chosen values of the elements of a matrix T.

Common affine transformation matrices

Affine transformations based on Eq. (2.6.–23).

Transformation Name	Affine Matrix, T	Coordinate Equations	Example
Identity	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x = v$ $y = w$	
Scaling	$\begin{bmatrix} c_x & 0 & 0 \\ 0 & c_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x = c_x v$ $y = c_y w$	
Rotation	$\begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x = v\cos\theta - w\sin\theta$ $y = v\cos\theta + w\sin\theta$	
Translation	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ t_x & t_y & 1 \end{bmatrix}$	$x = v + t_x$ $y = w + t_y$	
Shear (vertical)	$\begin{bmatrix} 1 & 0 & 0 \\ s_v & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x = v + s_v w$ $y = w$	
Shear (horizontal)	$\begin{bmatrix} 1 & s_h & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x = v$ $y = s_h v + w$	

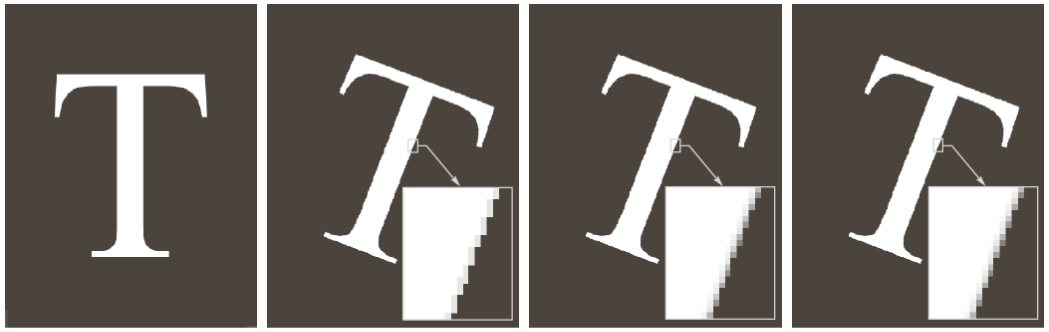
Effect of interpolation method on geometric transformation (rotation of 21º) using inverse mapping

(a) Original image

(b) rotation / interpolation using nearest neighbor interpolation

(c) rotation / interpolation using bilinear interpolation

(d) rotation / interpolation using bicubic interpolation



jagged edges

a b c d

Image registration

(used in image comparison)

Different images taken for the same scene

At different times (satellite image for monitoring earth movement)

With different viewing angles/ distances

…

Required: Aligning the images so that measurements and comparison could be made

In image registeration, Input and output images are known and transformation matrix T is unknown.

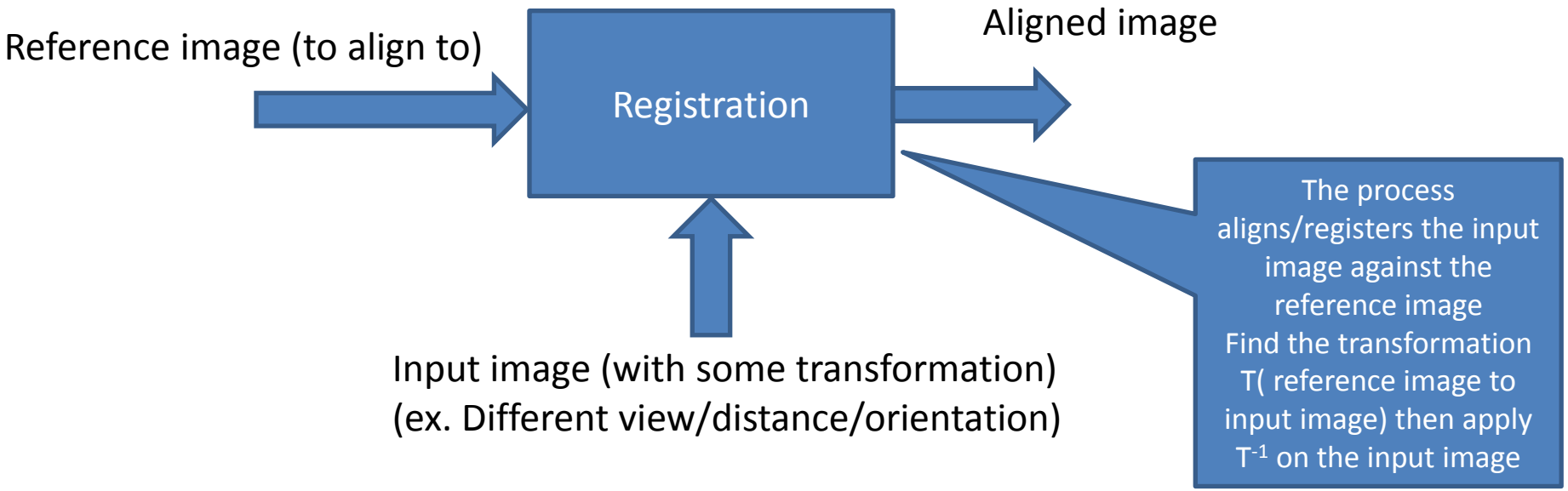


Image registration

A simple model for registration

$$x = c_1 v + c_2 w + c_3 vw + c_4$$

$$y = c_5 v + c_6 w + c_7 vw + c_8$$

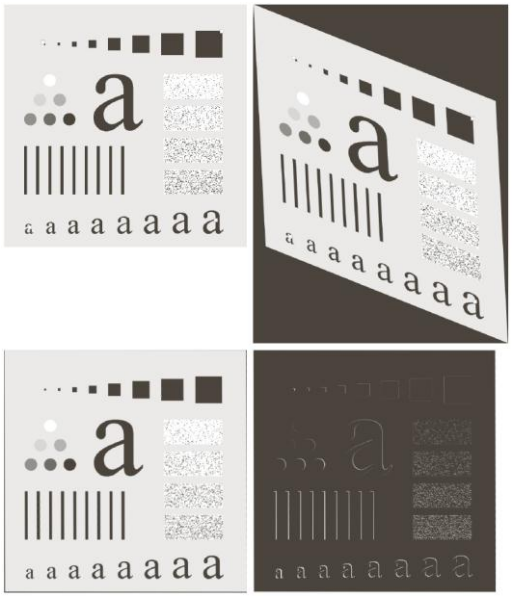
where x, y are refrence coordinates

and v, w are the input image coordinates

Apply these two equations at four **tie** points to obtain 8 equation in eight unknowns, solve them for the unknowns=> you got the model

Use the model to transform all the input image to the reference image

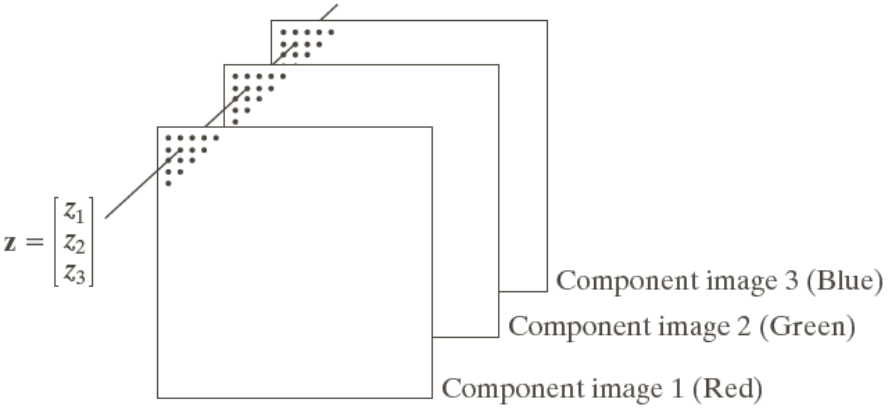
After transforming the coordinates, interpolation is used to assign intensity values



(a) Reference image
(b) input image
(c) Registered image
(d) Difference between a,c indicating some errors due to registration

Vector and matrix operations

- Color image can be represented as three image components, each in a different spectrum, for the same scene
- In such multispectral image, each pixel is represented by a vector in n-dimensions space (3D in RGB case)
- Vector theory could be applied on such images, for example, the distance between a pixel vector and a point in the RGB “D space is given by the shown equation
- The RGB 3D space representation can be generalized for n dimensions space (ex multispectral images of NASA LANDSAT



Euclidean distance between a pixel z and a point a :

$$D(z,a) = \left[(z-a)^T (z-a)\right]^{\frac{1}{2}}$$

$$= \left[(z_R - a_R)^2 + (z_G - a_G)^2 + (z_B - a_B)^2\right]^{\frac{1}{2}}$$

$$= \|z-a\|$$

Linear transformation and linear process

- N dimensions space multispectral image is transformed into m dimensions space by applying a linear transformation $w=A(z-a)$ where
  - z is the n by 1 pixel vector in the input image(vector for each pixel)
  - a is n by 1 vector
  - A is m by n transformation matrix
  - w is the resulting pixel vector in the m dimensions space
- Linear transformation has many applications in image processing

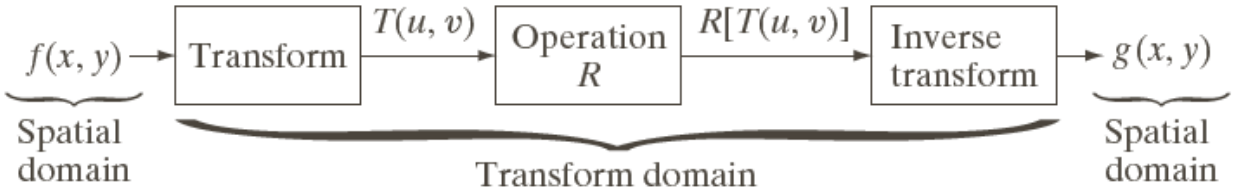
- A gray scale image is represented by a vector of MN by 1 by putting the image intensities in this vector row by row
- A linear process on such image is represented as $g=Hf+ń$, where
  - f is MN by 1 vector (one vector for the image)
  - H is MN by MN operation (process) matrix
  - ń is an MN by 1 noise vector
- Linear processes has many applications in image processing

Report discussion

Last lecture report: Problem 2.22

New report: Problem 2.26

Image transform



Linear two-dimensions general transform

$$T(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) r(x,y,u,v)$$

and its inverse transform is

$$f(x,y) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} T(u,v) s(x,y,u,v)$$

Transform Pair

where r(), s() are the forward and inverse transformation kernels

if

$$r(x,y,u,v) = r_1(x,u) r_2(y,v)$$

The kernel is separable

in addition, the kernel is symmetric

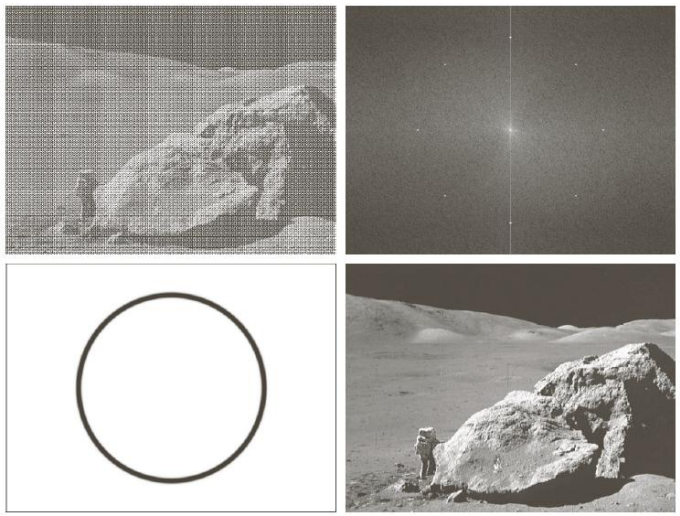
if r_1, r_2 are functionally equals

Note that each transformed pixel is calculated from all pixels in the input image and each inverse transformed pixel is calculated from all pixels in the transformed image

Example for linear two dimensions transform; Fourier transform

$$T(u,v) = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)e^{-j2\pi(\frac{ux}{M}+\frac{vy}{N})}$$

$$f(x,y) = \sum_{u=0}^{M-1}\sum_{v=0}^{N-1} T(u,v)e^{j2\pi(\frac{ux}{M}+\frac{vy}{N})}$$



a b
c d

(a) Image corrupted by sinusoidal interference. (b) Magnitude of the Fourier transform showing the bursts of energy responsible for the interference. (c) Mask used to eliminate the energy bursts. (d) Result of computing the inverse of the modified Fourier transform. (Original image courtesy of NASA.)

Transform matrix notation

If

1- both the forward and the reverse kernels are separable and symmetric

2- The operated image is square M by M

The transform can be put in matrix notation as follows

$T=AFA$, F is an M x M matrix containing the elements of the image f(x,y),

Where T is the result , A is M by M matrix whose elements $a_{jj}=r_1(i,j)$

to obtain the inverse transform,

Multiplying by B from both sides

$BTB=BAFAB$

If $B=A^{-1}$, then the inverse transform is calculated as

$F=BTB$, where Matrix F contains the elements(pixels value) of the image f(x,y).

indicating that image f(x,y) can be recovered completely from its forward transform(T).If B is not equal to the inverse of A, you can get an approximation for f(x,y):

$\hat{F}=BAFAB$

Probabilistic models

if intensity values are treated as random quantities

Possible intensities $z_i, i = 0,1,2,..., L-1$

$$P(z_k) = \frac{\text{n. z}_\text{k} \text{ happened}}{\text{MN}}$$

$$\sum_{k=0}^{L-1} p(z_k) = 1$$

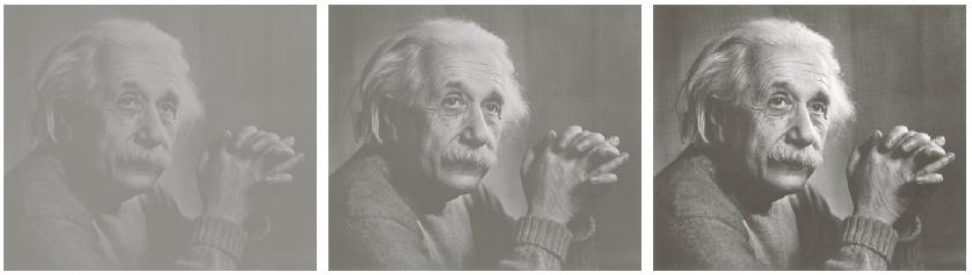
Spread of pixel intensities arround their mean

$$\sigma^2 = \sum_{k=0}^{L-1} (z_k - m)^2 p(z_k) = \text{var}\,iance$$

$$\sqrt{\text{var}\,iance} = \sigma = \text{standard deviation}$$

variance and standard deviation are measures for image contrast

mean of an image=m=???



a b c

Image	SD	Variance	Description
a	14.3	204.3	Low contrast
b	31.6	997.8	Medium contrast
c	49.2	2424.9	Heigh contrast

Probabilistic models

$$\sigma^2 = \sum_{k=0}^{L-1} (z_k - m)^2 \, p(z_k) = \text{var}\,iance = \text{Second moment}$$

In General the nth moment is calculated as

$$\sigma^n = \sum_{k=0}^{L-1} (z_k - m)^n \, p(z_k)$$

There are some relation to the moment value and pixel values in an image
Positive third moment=> pixel values have bias to values higher than the mean
Negative third moment=>pixel values have bias to values smaller than the mean
Zero third moment=> pixel values are distributed uniformly the mean

Level of mathematics needed
Treating pixel intensity as random variable: random variable theory
Sequence of image whose pixel intensities are treated as random variable :
Stochastic process theory

Lecture scope

In this lecture, we covered from section 2.6.5 to to the end of chapter 2 in the book: Digital image processing 3ED, By Gonzalez and Woods
Workout exercises:)